CE143: COMPUTER CONCEPTS & PROGRAMMING

UNIT-1
Introduction to ‘C’ language

N. A. Shaikh

nishatshaikh.it@charusat.ac.in

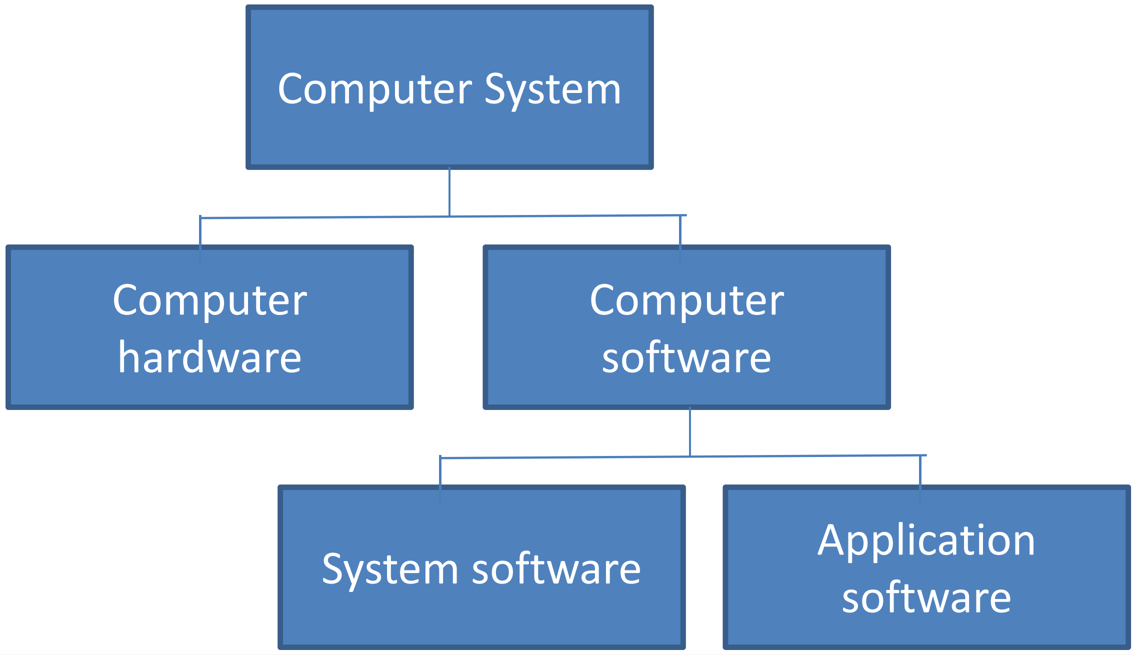
Topics to be covered

- Program
- Software
- Instruction
- Debugging
- Compilation and Execution of C Program
- Difference between Header files & library files
- Compiler and Interpreter
- Procedure Oriented Language
- Importance of C
- Basic structure of C
- Algorithms & Flowchart

Introduction to Computer

A **computer** is an electronic device that accept data(input) and, process data arithmetically and logically, produce information(output).

- It is divided into two main categories
  - Hardware
  - Software



Hardware & Software

Hardware refers to the physical elements of a computer.

Ex: keyboard, monitor, mouse, CPU etc..

Software refers to the set of instruction that tells a computer what to do or how to perform a task.

Ex: Ms word, excel, power point, spread sheets etc..

Types of Software:

- System Software
- Application Software

Types of Software

System software controls a computer’s internal functioning, chiefly through an operating system, and also controls such peripherals as monitors, printers, and storage devices

Ex: Operating Systems, Compiler, Loader, Linker, Interpreter etc..

Application software directs the computer to execute commands given by the user

Ex: games, spreadsheets, word processor, database, web browsers etc..

Language

A **Language** is a medium of communication which has it's own vocabulary and grammar.

Human Language: Commonly used to express feelings and understand other person expressions. It can be oral or gestural kind of communication.

Computer Language: Computer languages are the languages by which a user command a computer to work on the algorithm which a user has written to get an output.

Instruction

A sentence formed by using a programming language or we can say a sentence written in a programming language is called an **Instruction**.

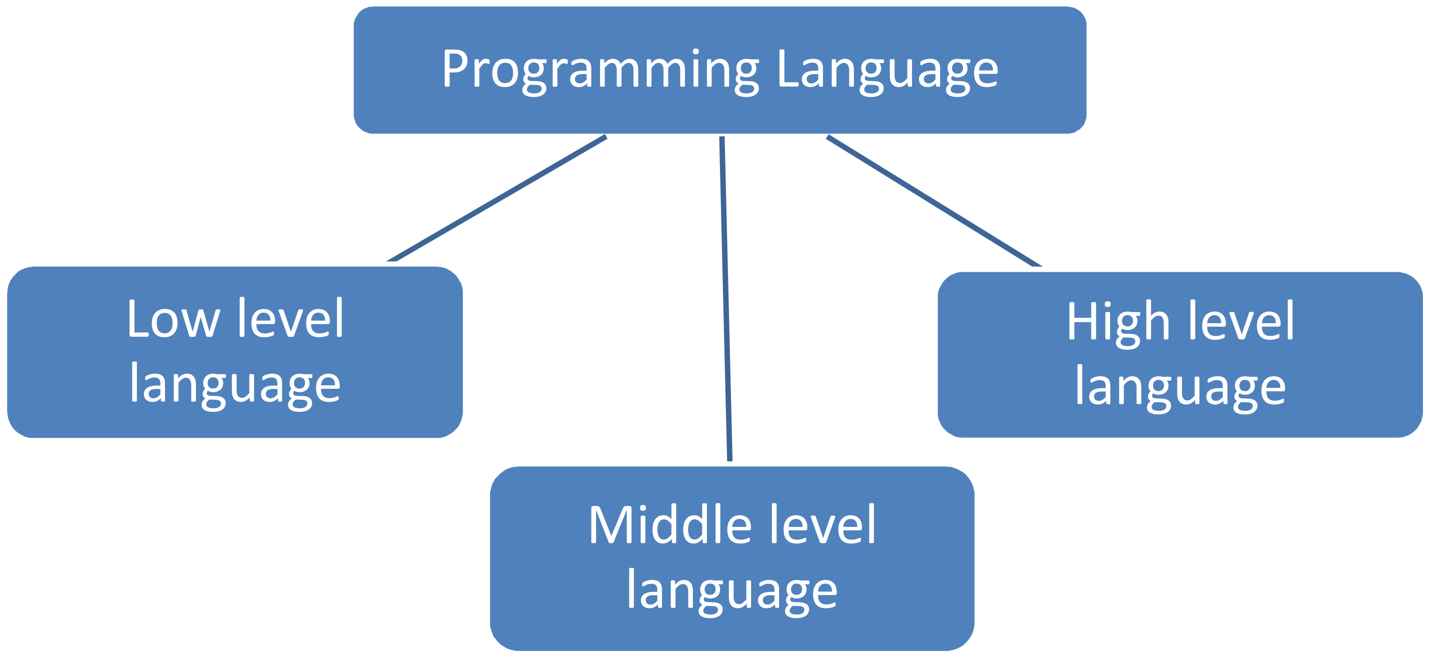
Program

A set of Instructions organized in sequence to perform a certain task or to achieve a given objective , is called **program**.

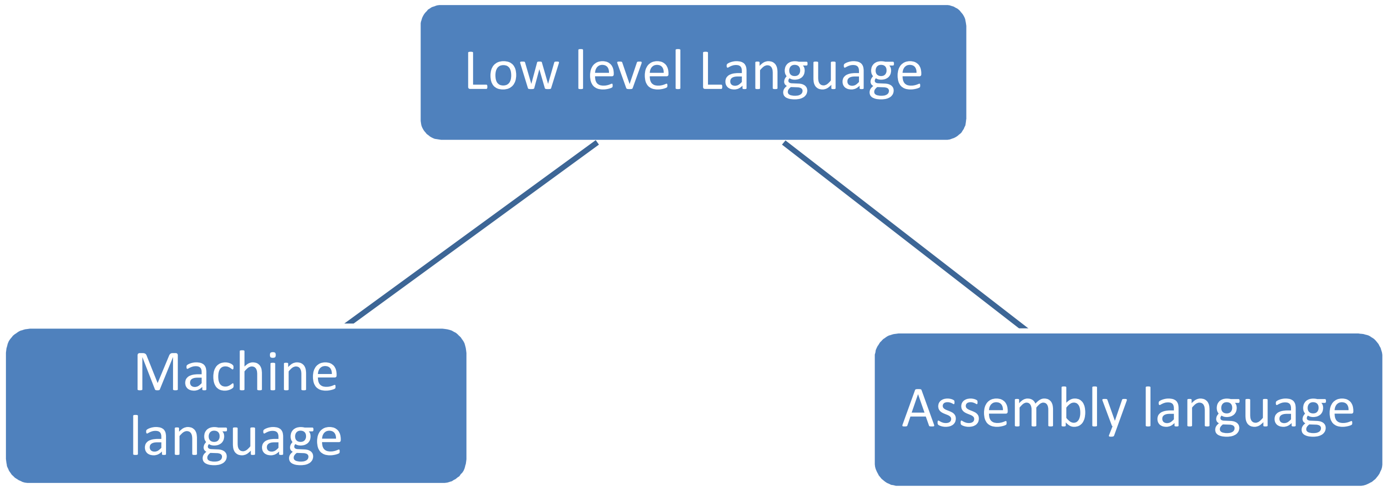
The process of writing a program is called **programming**.

A language used to create computer programs is called **programming language**.

Programming language



Low Level Language



(Binary Language)
Use 1's & 0's to create instructions

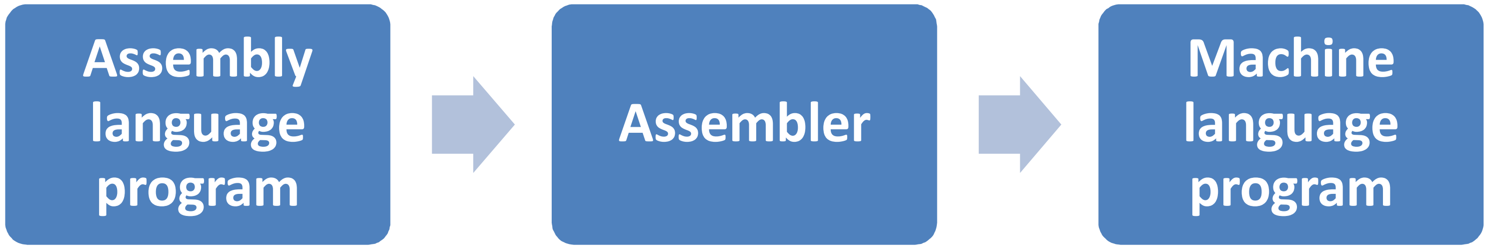
Ex: 11001 001010 000101

Use mnemonics to create instructions

Ex: ADD 10 5

Assembler

Software that translates as assembly language program into an equivalent machine language program of a computer is known as **Assembler**



High Level Language

High level Language

Similar to human Language

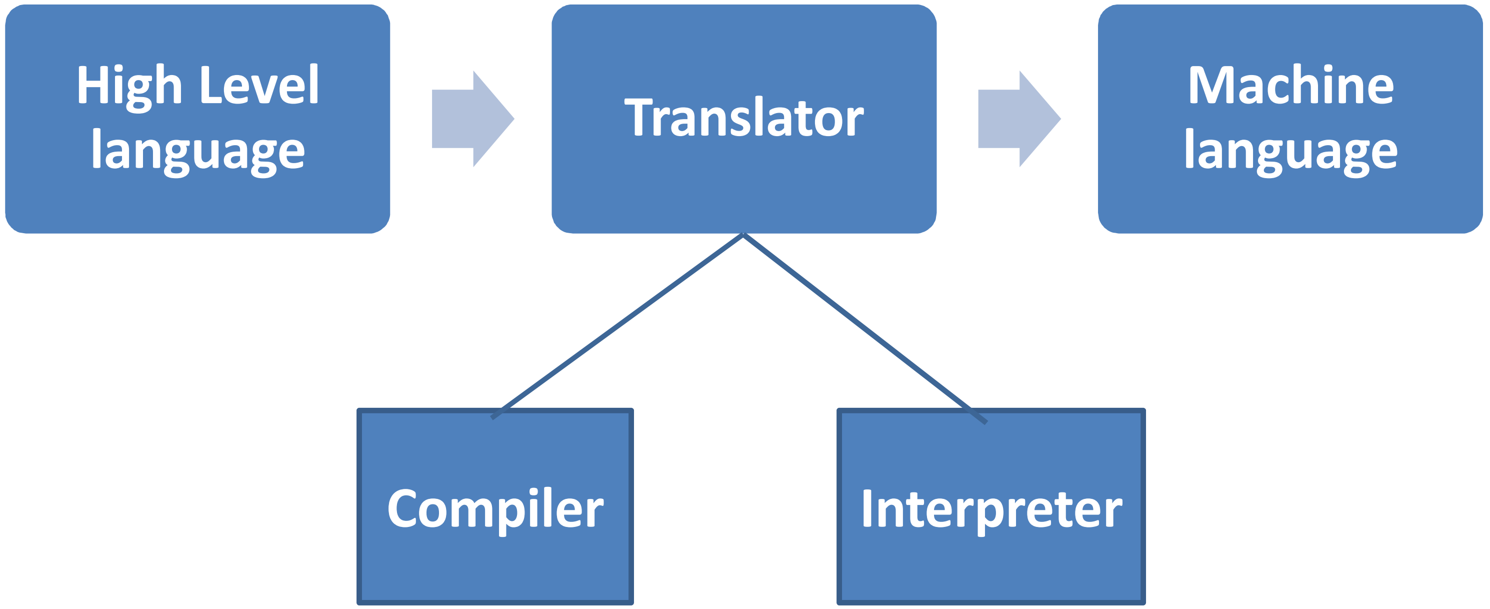
COBOL, FORTRAN, PASCAL, C#, PROLOG, JAVA, Python , .NET etc

Ex:

```
c=a + b;
return c;
```

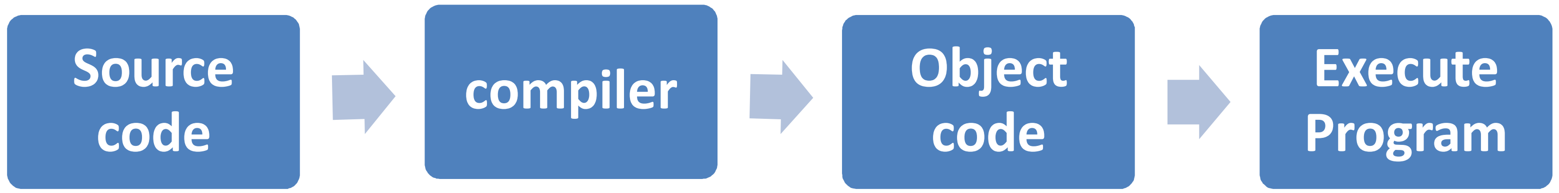
Translator

A **translator** is software that converts the instructions written in some programming language into the form (binary) which is understandable by computer.

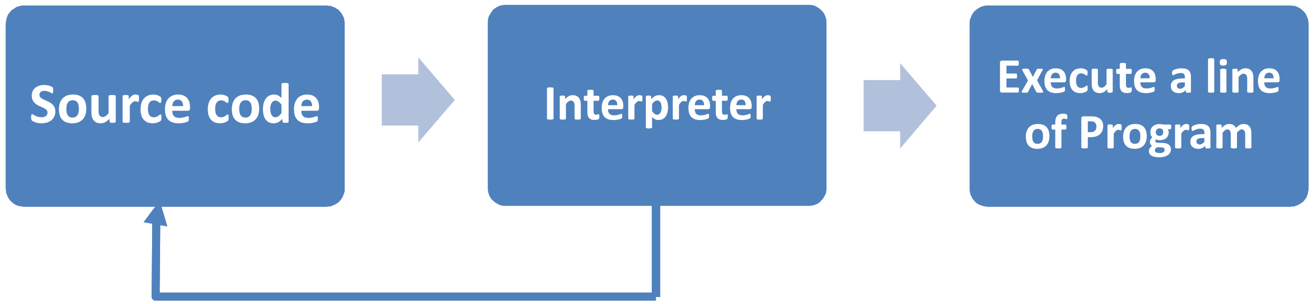


Compiler & Interpreter

Using Compiler



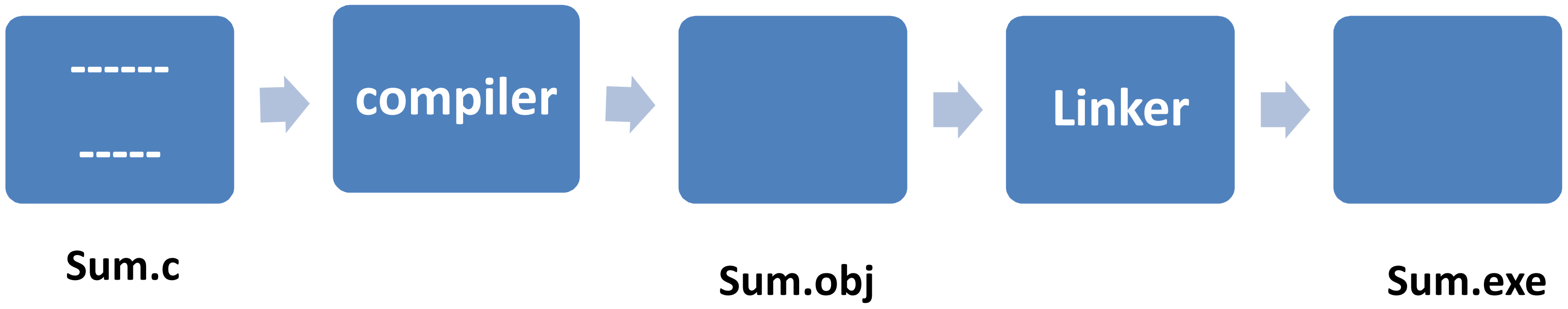
Using Interpreter



Compiler VS Interpreter

Compiler	Interpreter
Compiler takes Entire program as input	Interpreter takes Single instruction as input.
Intermediate Object code is generated	No intermediate Object code is generated
Memory requirement: More (As object code is generated)	Memory requirement: Less
Display all errors after compilation, all at the same time.	Displays error of each line one by one(if any)
Programming languages like C, C++, Java use compilers.	Programming languages like JavaScript, Python, Ruby, PHP use interpreters.
The compilation is done before execution.	Compilation and execution take place simultaneously.
Comparatively faster	Slower
Error detection: Difficult	Error detection: Easier comparatively

Linker



- It **links all the functions and files** required by the object code and **converts the object code to executable code**.
- The converted code is stored with **a .exe** extension.
- The linker gives **error** if the file or function that has to be linked does not exist.

Loader

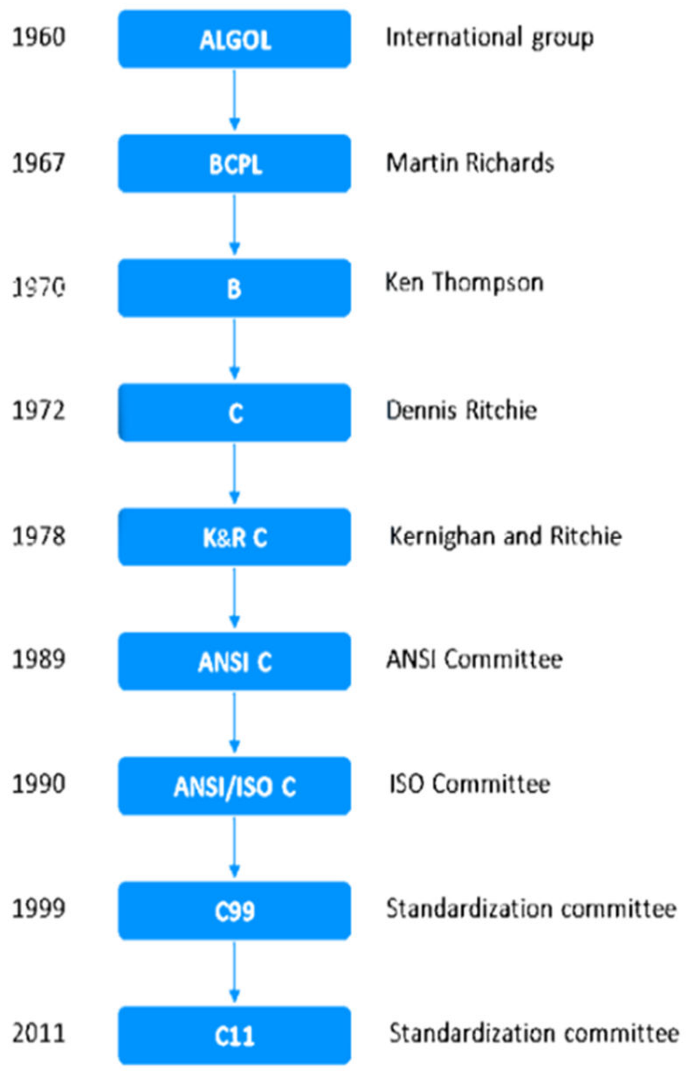
- It is a **part of an operating system** that is responsible for **loading programs.**
- It places programs into memory and prepares them for execution
- In this stage can also generate errors. These error are called **runtime error**.

Middle Level Language

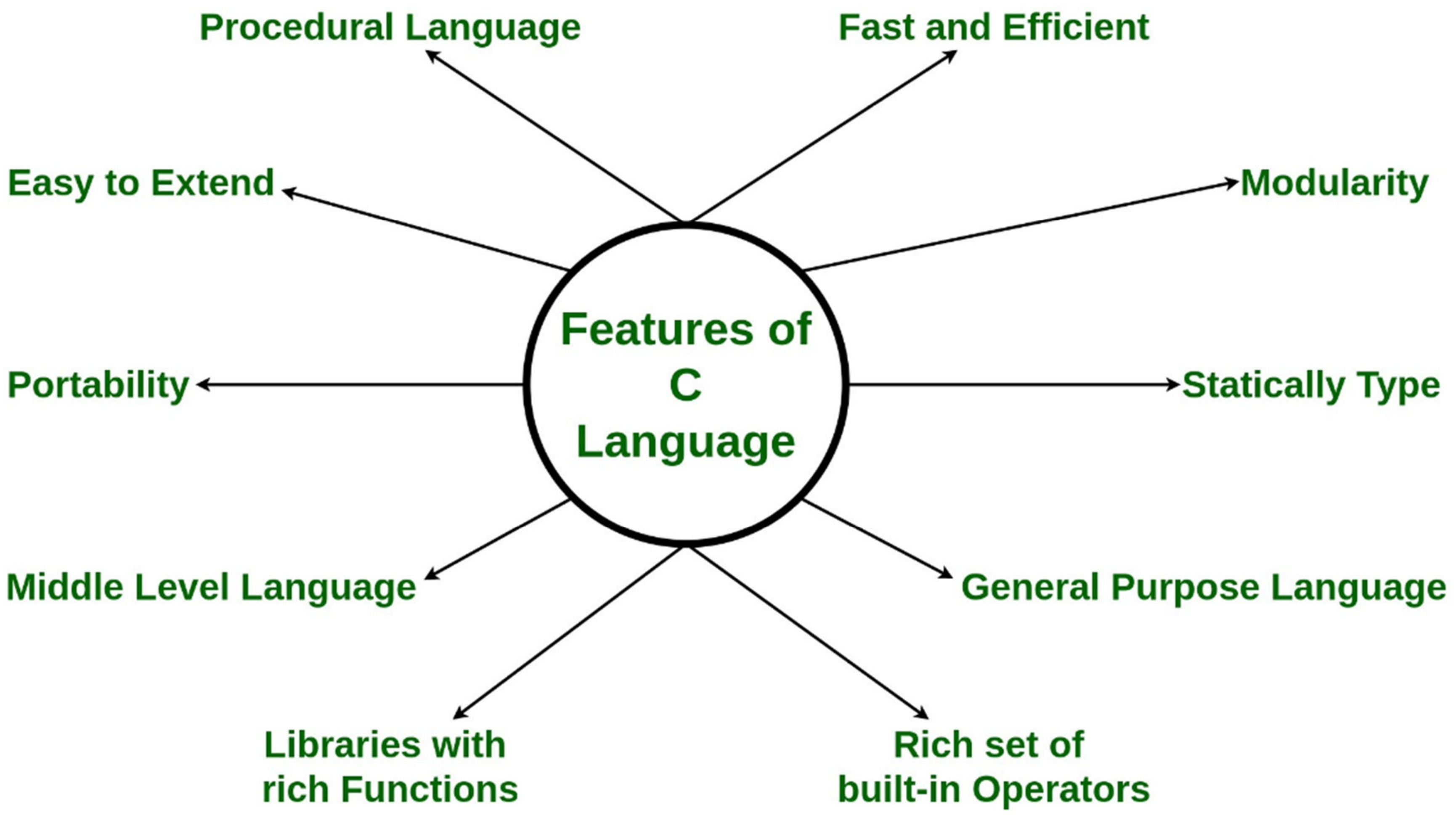
- Middle level language incorporates the features of both **high level languages** and **low level languages**.
- It uses both the English words as well as low level instructions(including binary digits also)
- **Ex: C Language**
  - By using the C language, the user is capable of doing the system programming for writing operating system as well as application programming

History of C

- C is a programming language which born at “AT & T’s Bell Laboratory” of USA in 1972.
- C was written by Dennis Ritchie, that is why he is also called as father of c programming language.
- As many of the features were derived from “B” Language that is why it was named as “C”.
- After 7-8 years C++ came into existence which was first example of object oriented programming .



Features of C



Algorithms

A typical programming task can be divided into two phases:

Problem solving phase

- produce an ordered sequence of steps that describe solution of problem
- this sequence of steps is called an **algorithm**

Implementation phase

- implement the program in some programming language
- Program is implementation of ALGORITHM

Algorithms

“An Algorithm is defined as Sequence of steps to solve the problems”

Algorithm has the following characteristics

- **Input:** An algorithm may or may not require input
- **Output:** Each algorithm is expected to produce at least one result
- **Definiteness:** Each instruction must be clear and unambiguous.
- **Finiteness:** algorithm must terminate after finite number of steps
- **Effectiveness:** Every step must be basic and essential.

Algorithms

Algorithm to establish a telephonic communication between two subscribers:

(i) Dial a phone number

(ii) Phone rings at the called party

(iii) Caller waits for the response

(iv) Called party picks up the phone

(v) Conversation begins between them

(vi) After the conversation, both disconnect the call.

Algorithms

Algorithm to find the sum of two numbers:

Step 1: Start

Step 2: Input two numbers say A & B

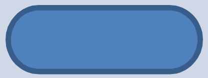
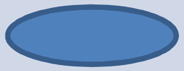
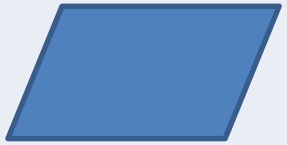
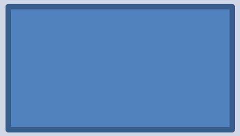
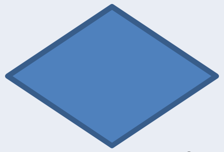
Step 3: SUM = A + B

Step 4: Display SUM

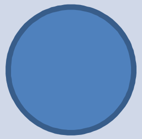
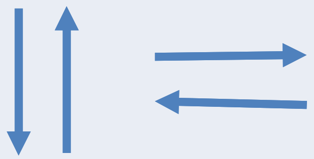
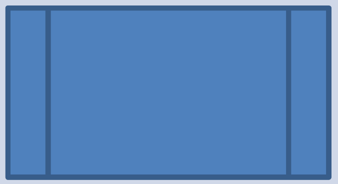
Step 5: Stop

Flowcharts

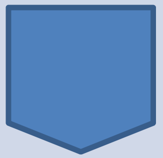
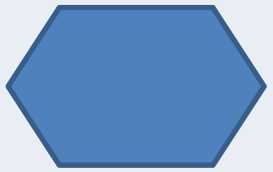
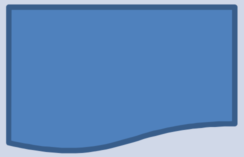
Flowchart is a diagrammatic/Graphical representation of any algorithm

Name	Symbol	Purpose
Terminal	 Terminator  oval	Start/stop/begin/end
Input / Output	 parallelogram	Input / Output of data
Process	 Rectangle	Used for arithmetic operations and data-manipulations
Decision box	 Diamond	Decision making. Used to represent the operation in which there are two/three alternatives, true and false etc

Flowcharts

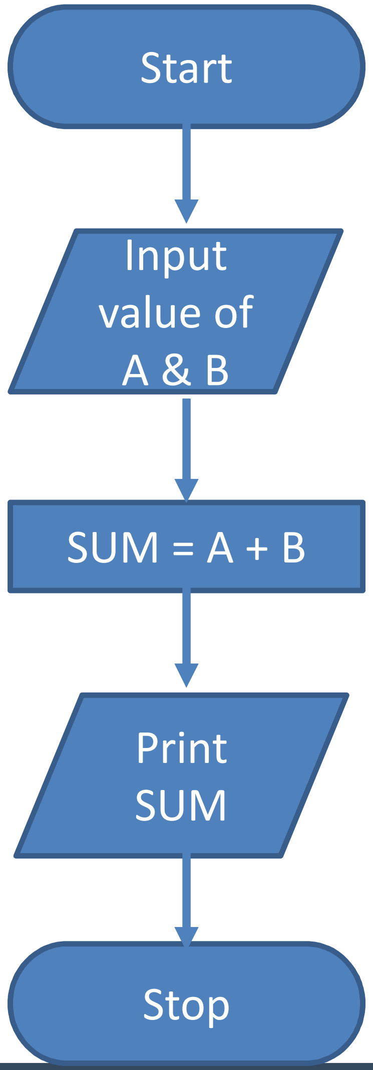
Name	Symbol	Purpose
Connector	 circle	Used to connect different parts of flowchart
Flow	 Arrows	Joins 2 symbols and also represents flow of execution
Pre defined process	 Double sided rectangle	Function Used to represent a group of statements performing one processing task.

Algorithms & Flowcharts

Name	Symbol	Purpose
Off Page connector	 pentagon	Used to connect flowchart in 2 different pages
For loop symbol	 Hexagon	Shows initialization, condition and incrementation of loop variables
Document	 Print out	Shows the data that is ready for print out

Algorithms & Flowcharts

Flowchart to find the sum of two numbers



Algorithms & Flowcharts

Algorithm & Flowchart to convert temperature from Celsius to Fahrenheit

C : temperature in Celsius

F : temperature Fahrenheit

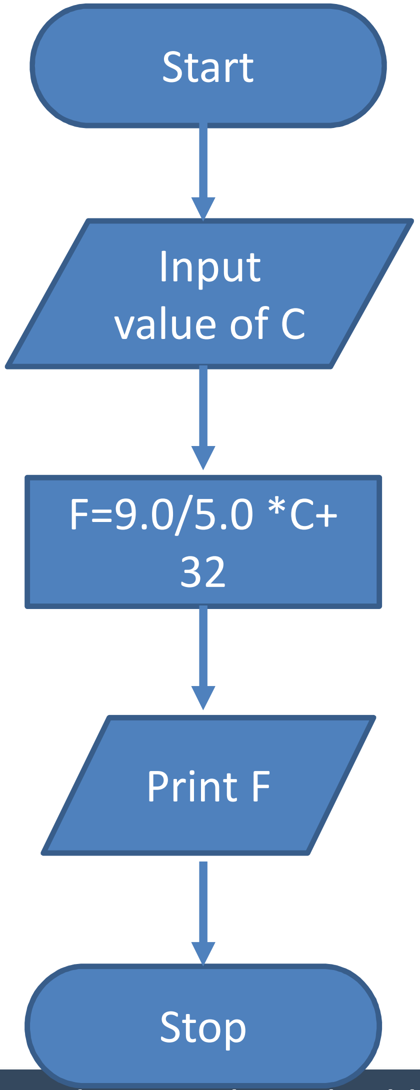
Step 1: Start

Step 2: Input temperature in Celsius say C

Step 3: F = (9.0/5.0 x C) + 32

Step 4: Display Temperature in Fahrenheit F

Step 5: Stop



Algorithms & Flowcharts

Exercise-1: Write Algorithm & Draw Flowchart to convert temperature from Fahrenheit to Celsius

Algorithms & Flowcharts

Algorithm & Flowchart to find Area and Perimeter of Square

L : Side Length of Square

AREA : Area of Square

PERIMETER : Perimeter of Square

Step 1: Start

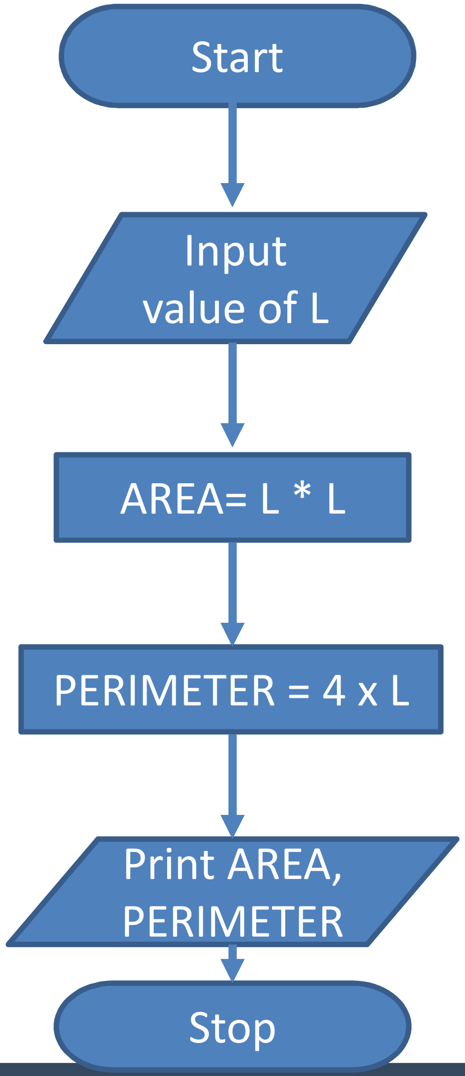
Step 2: Input Side Length of Square say L

Step 3: Area = L x L

Step 4: PERIMETER = 4 x L

Step 5: Display AREA, PERIMETER

Step 6: Stop



Algorithms & Flowcharts

Exercise-2: Algorithm & Flowchart to find Area and Perimeter of Rectangle

Exercise-3: Algorithm & Flowchart to find Area and Perimeter of Circle

Exercise-4: Algorithm & Flowchart to find Area & Perimeter of Triangle

Algorithms & Flowcharts

Algorithm & Flowchart to find Simple Interest

P : Principle Amount

N : Time in Years

R : % Annual Rate of Interest

SI : Simple Interest

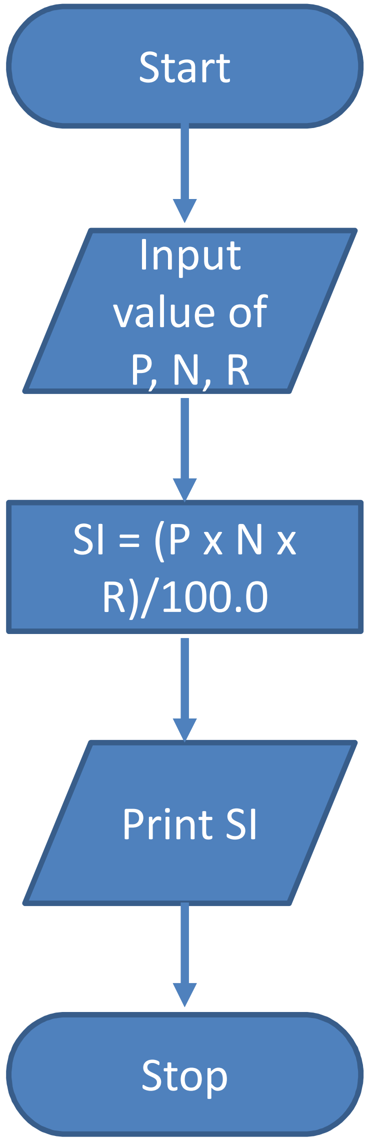
Step 1: Start

Step 2: Input value of P, N, R

Step 3: SI = (P x N x R)/100.0

Step 4: Display SI

Step 5: Stop



Algorithms & Flowcharts

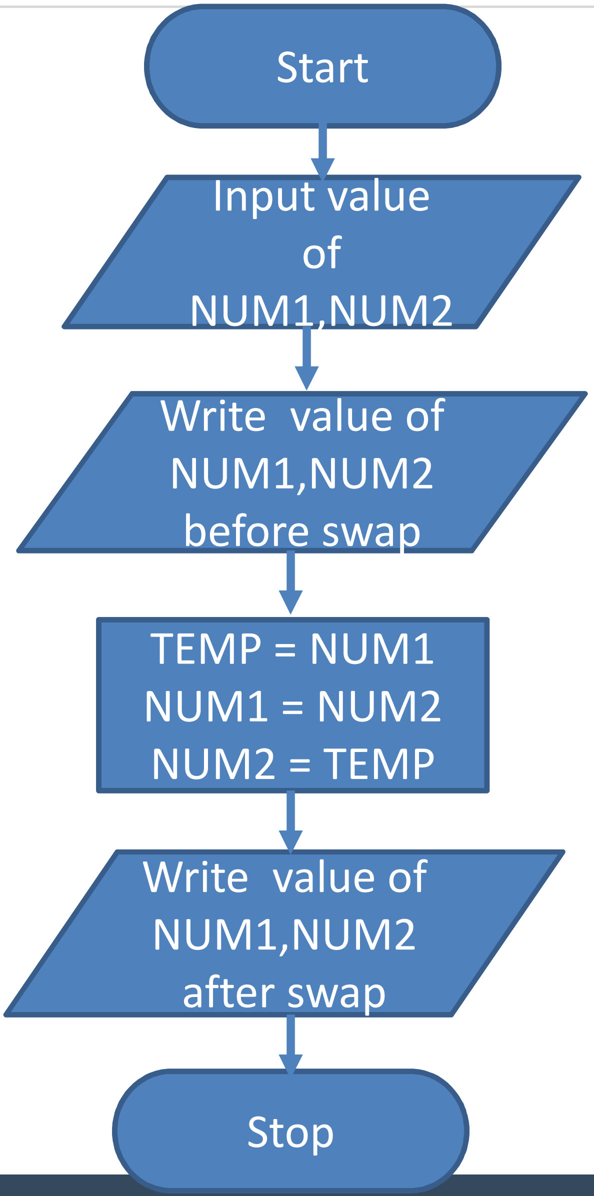
Execise-5:Write Algorithm & Draw Flowchart to find Compound Interest

Algorithms & Flowcharts

Algorithm & Flowchart to Swap Two Numbers using Temporary Variable

```
Step 1: Start
Step 2: Input Two Numbers Say NUM1,NUM2
Step 3: Display Before Swap Values NUM1, NUM2
Step 4: TEMP = NUM1
Step 5: NUM1 = NUM2
Step 6: NUM2 = TEMP
Step 7: Display After Swap Values NUM1,NUM
Step 8: Stop
```

Algorithms & Flowcharts


Start
Input value of NUM1,NUM2
Write value of NUM1,NUM2 before swap
TEMP = NUM1
NUM1 = NUM2
NUM2 = TEMP
Write value of NUM1,NUM2 after swap
Stop

Algorithms & Flowcharts

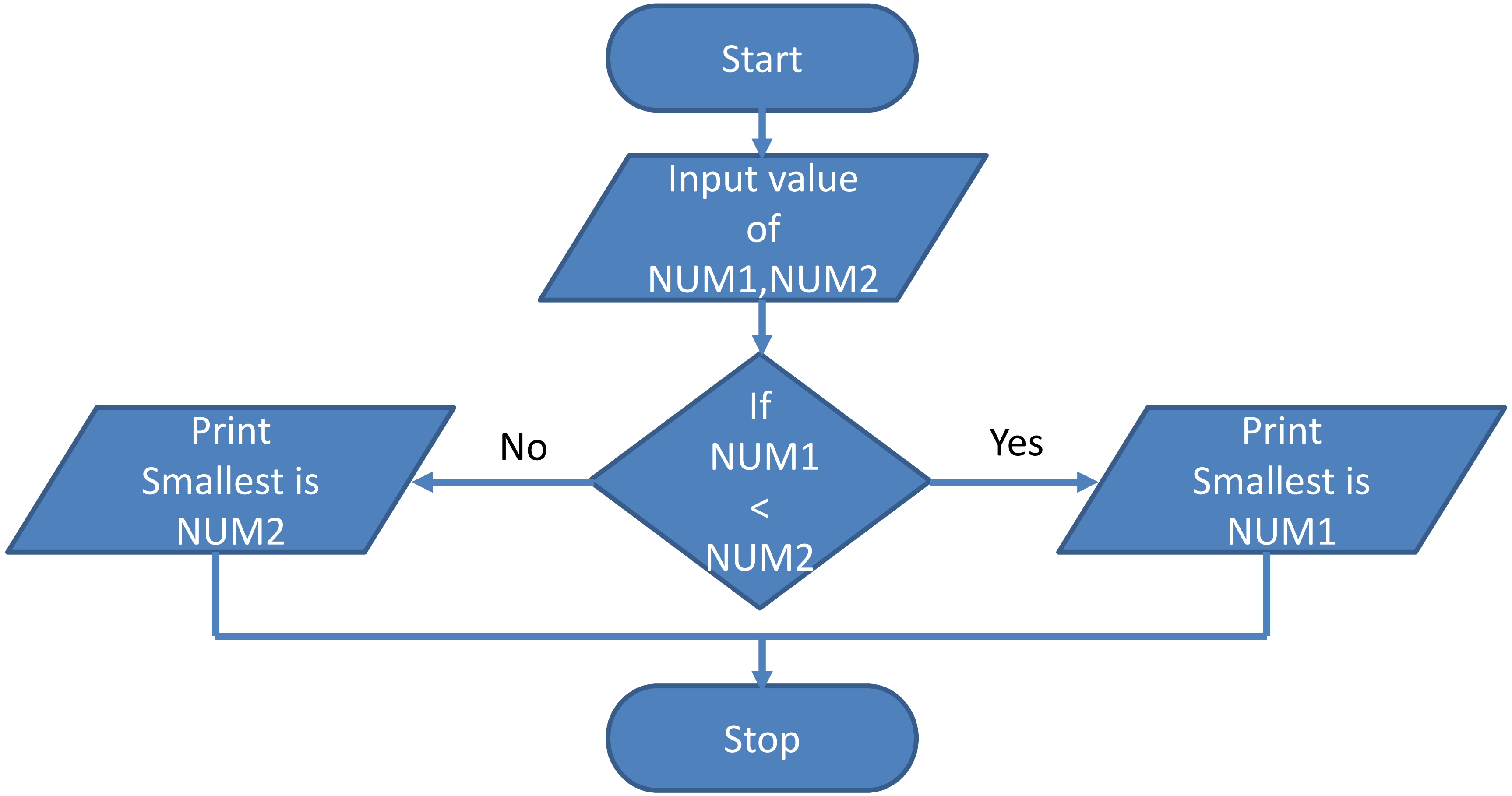
Exercise-6:Write Algorithm & Draw Flowchart to Swap Two Numbers without using temporary variable

Algorithms & Flowcharts

Algorithm & Flowchart to find the smallest of two numbers

```
Step 1: Start
Step 2: Input two numbers say NUM1,NUM2
Step 3: IF NUM1 < NUM2 THEN
                    print smallest is NUM1
               ELSE
                    print smallest is NUM2
              ENDIF
Step 4: Stop
```

Algorithms & Flowcharts



Algorithms & Flowcharts

Exercise-7:Write Algorithm & Draw Flowchart to find the largest of two numbers

Exercise-8:Write Algorithm & Draw Flowchart to find the largest of three numbers

Pseudocode

Pseudocode is an informal way of programming description that does not require any strict programming language syntax or underlying technology considerations

Pseudocode

Write a pseudocode to find sum and average of given two numbers.

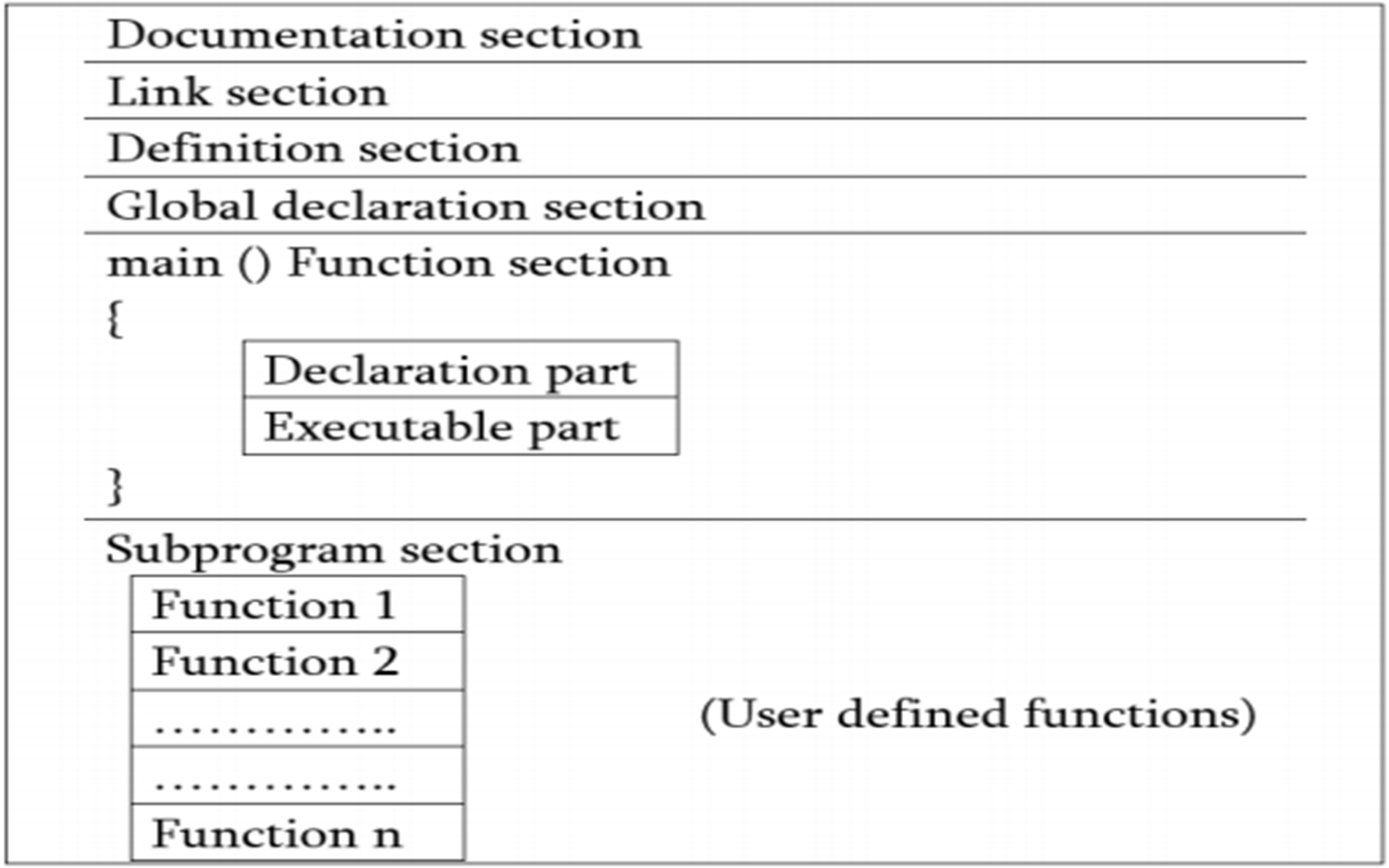
```
Begin
WRITE “Please enter two numbers to add”
READ num1
READ num2
Sum = num1+num2
Avg = Sum/2
WRITE Sum, Avg
End
```

Pseudocode

Write down Pseudocode that will take marks of physics, chemistry and math as input, calculates the average and displays output.

```
Begin
PRINT “please enter marks of physics”
INPUT Phy_marks
PRINT “please enter marks of chemistry”
INPUT Chem_marks
PRINT “please enter marks of maths”
INPUT math_marks
Avg = (Phy_marks + Chem_marks + math_marks)/3
PRINT Avg
End
```

Basic Structure of C Program



Documentation Section

Gives the details associated with the program and overview of the code

- Program name,
- the details of the author
- the time of coding and
- description

// Single line comment

/* Multi-line comment */

Example

```
/*
File Name: Helloworld.c
Author: Prof. Nishat Shaikh
date: 11/11/2020
description: a program to display hello world
*/
```

Link Section

- **Used to declare all the header files that will be used in the program.**
- **This leads to the compiler being told to link the header files to the system libraries.**

Example:

- #include<stdio.h>   //Standard Input / Output
- #include<conio.h>   //Console Input / Output
- #include<math.h>   //mathematical operations (sqrt(), pow() etc..)

#include (Preprocessor directive)

- Preprocessor directives start with #
- #include copies a file into the source code

Link Section

stdio.h (Header File)

Header File	Library File
They have the extension .h	They have the extension .lib
They contain function declaration	They contain function definations
only have header name	Have actual implementation code of the header
Header files are human readable(in the form of source code)	Library files are non human readable(in the form of machine code)
included by using a command #include	included in last stage by special software called as linker.

Definition Section

Define different symbolic constants(Macros)

Example:

```
#define PI 3.14
#define TRUE 1
#define FALSE 0
```

- Should not end with a semicolon
- Generally written in uppercase to distinguished from lowercase variable names

Global Declaration Section

Global variables(visible/accessible throughout the program)and user defined functions are declared in this section.

Example:

```
#include<stdio.h>
int a=7;            //Global Declaration
int main()
{
--------
--------
}
```

main() function Section

Every C-Program must have exactly one main function

```
int main()
{
Statement 1;
Statement 2;
}
```

- **Declaration Part:** Declares all the variables used in the executable part.
- **Execution Part:** Main Logic

main() function Section

```
int main()
{
//Declaration
int number, principal ,rate ,interest;
// Execution
number=10;
principal=1000;
rate=20;
interest=(number*principal*rate)/100;
}
```

printf() function

- Used to print on standard output(screen)
- inbuilt library functions in C programming language
- Defined in “stdio.h”
- We have to include “stdio.h” file to make use of these printf()
- To generate a newline,we use “\n” in C printf() statement.

```
int main()
{
printf(“hello \n”);
printf(“How are you?”);
}
```

SubProgram Section

All the user-defined functions are defined in this section

Example:

```
int add(int a, int b)
{
return a+b;
}
```

First C Program

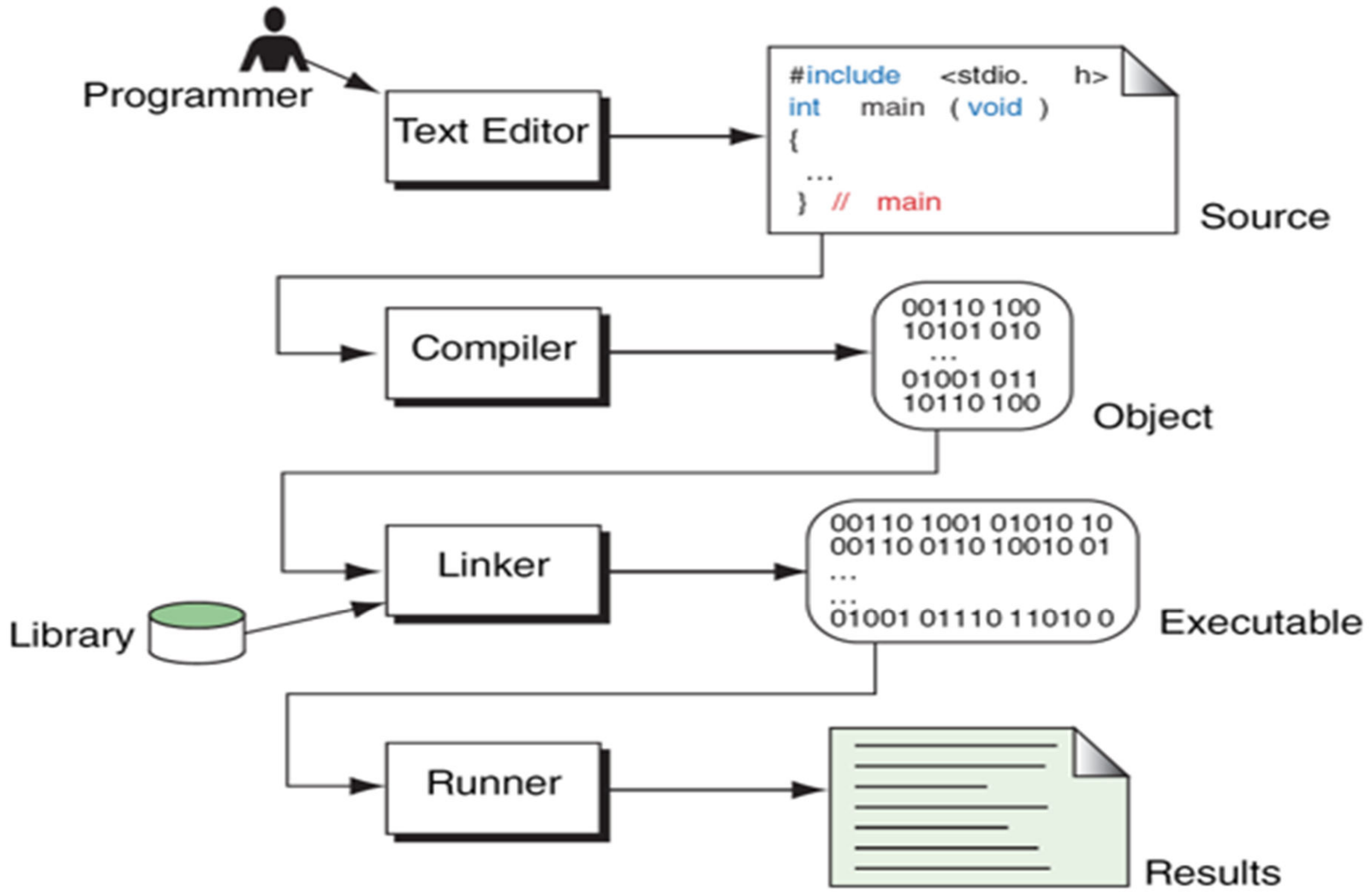
Code

```
#include<stdio.h>
int main()
{
printf("First C Program");
return 0;
}
```

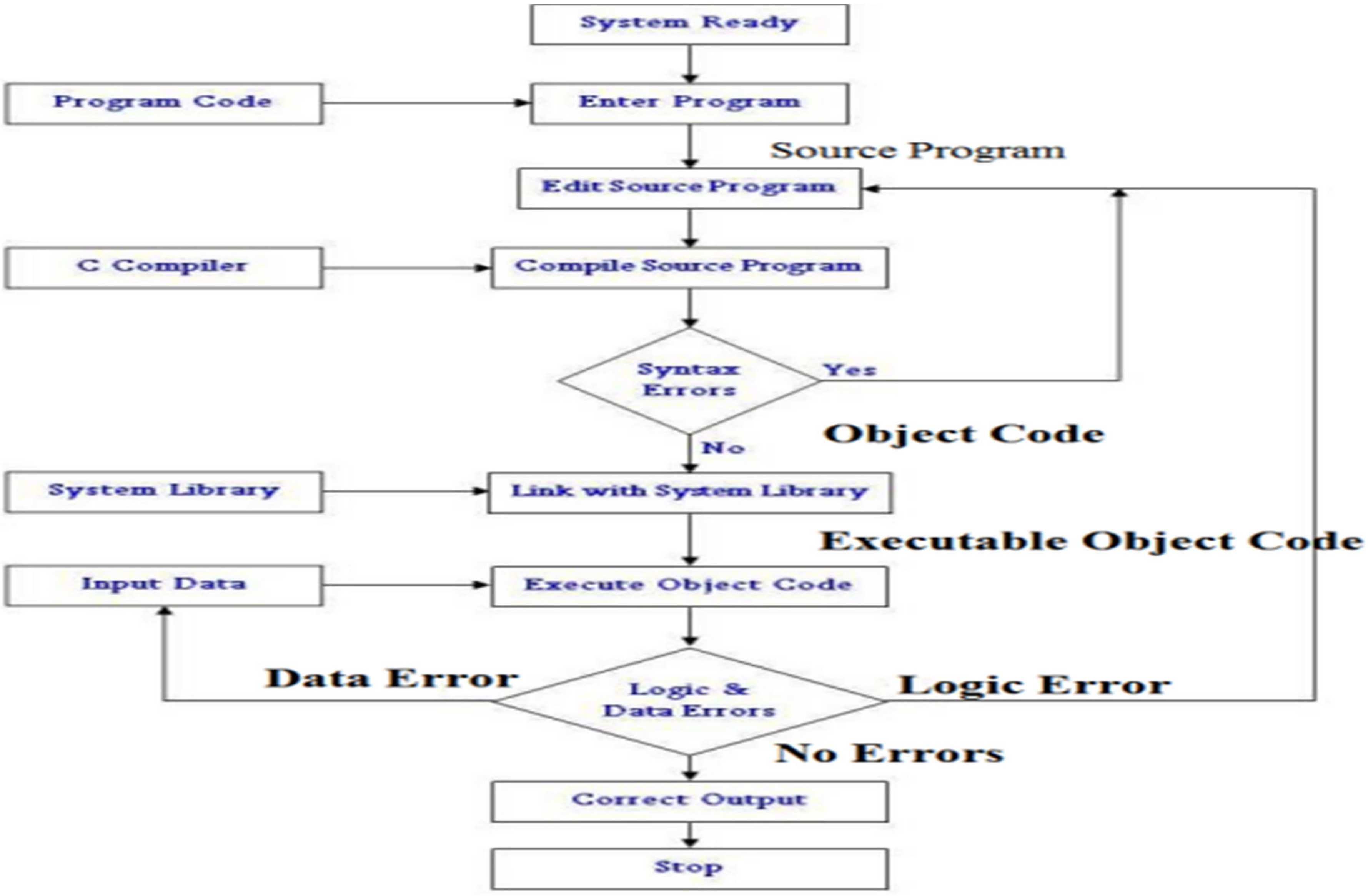
Output

```
First C Program
```

Compilation and Execution of C Program



Compilation and Execution of C Program

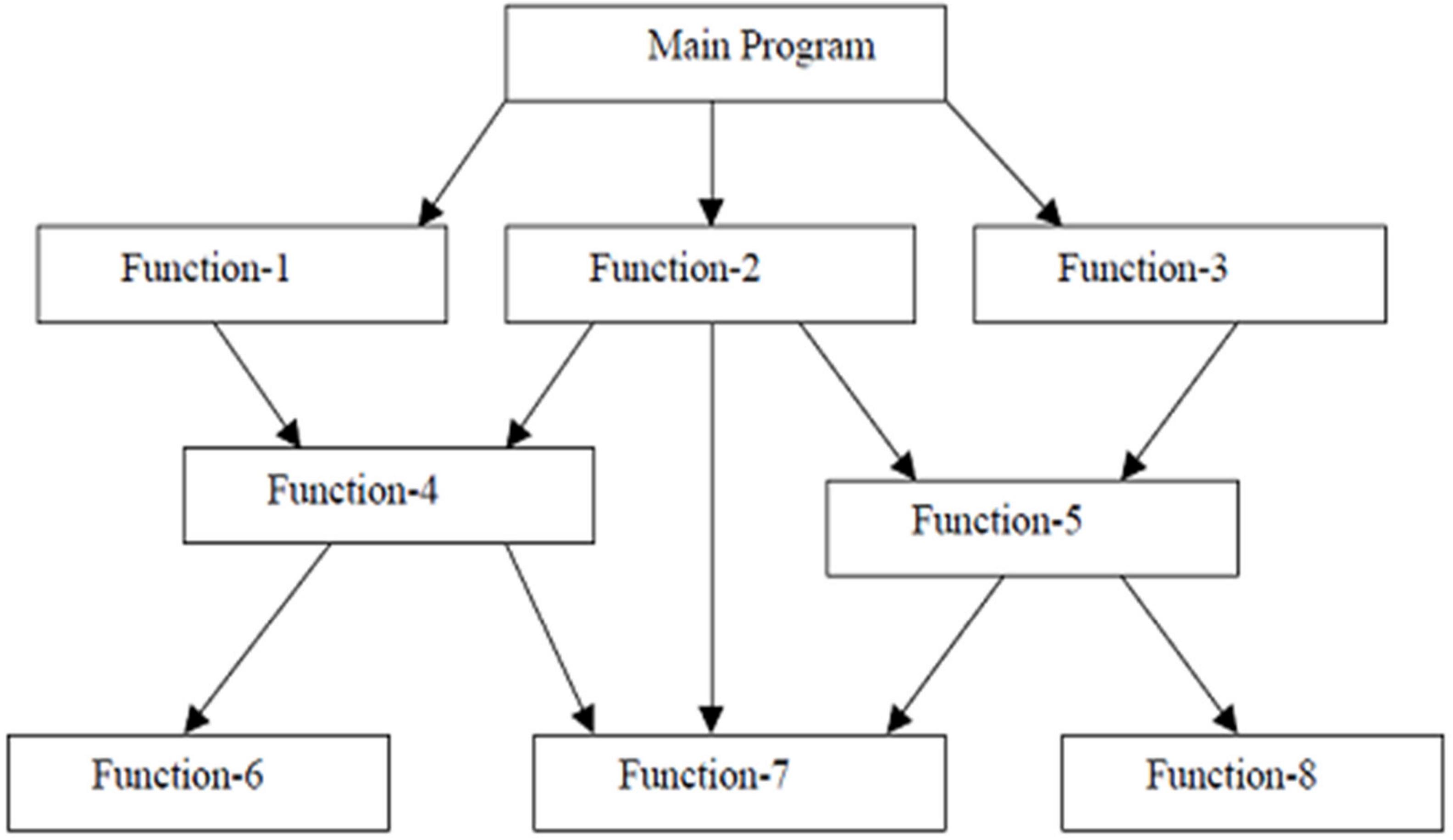


Procedure Oriented Programming(POP)

- It means "**a set of procedures**" which is a "**set of subroutines**" or a "**set of functions**".
- In POP, Programmer combines related sequence of statements into one single place, called procedure.
- When program become larger, it is divided into function & each function has clearly defined purpose.
- The primary focus is on functions
- This technique is also known as **Top-down** Programming

E.g.:- c, basic, FORTRAN.

Procedure Oriented Language(POP)



Structure of procedural oriented programs

Procedure Oriented Programming(POP)

OOP	POP
OOP stands for Object Oriented Programing.	POP stands for Procedural Oriented Programming.
OOP follows bottom up approach.	POP follows top down approach.
A program is divided to objects and their interactions.	A program is divided into funtions and they interacts.
Inheritance is supported.	Inheritance is not supported.
Access control is supported via access modifiers.	No access modifiers are supported.
Encapsulation is used to hide data.	No data hiding present. Data is globally accessible.
Main focus is on 'data security'. Hence, only objects are permitted to access the entities of a class.	Main focus is on "how to get the task done" i.e. on the procedure or structure of a program .
Example: C++, Java	Example: C, Pascal

End of Unit-01